ECEC 204: Design with Microcontrollers
Pulse Width Modulation

Prof. Naga Kandasamy
ECE Department
Drexel University

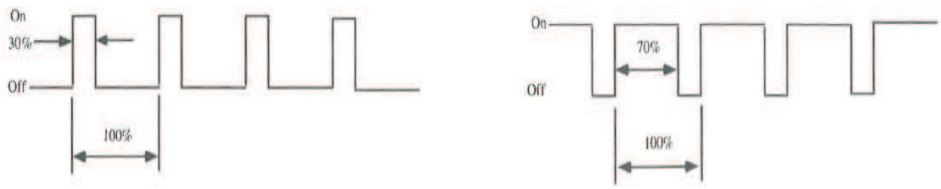
July 29, 2019

The notes have been derived from the following sources:

- D. Pack and S. Barrett, *Microcontroller Theory and Applications: HC12 and S12,* 2^{nd} Edition, Prentice Hall, 2007.
- M. Barr, "Introduction to Pulse Width Modulation," *Embedded Systems Programing*, September 2001.

The MSP432 microcontroller is equipped with a Pulse Width Modulation (PWM) system, a powerful technique for controlling analog circuits with a microprocessor's digital outputs. PWM is a way of digitally encoding analog signal levels and is employed in a wide variety of applications, ranging from measurement and communications to power control and conversion.

An analog signal has a continuously varying value, with infinite resolution in both time and magnitude. A nine-volt battery is an example of an analog device, in that its output voltage is not precisely 9V, changes over time, and can take any real-numbered value. Similarly, the amount of current drawn from a battery is not limited to a finite set of possible values. Analog signals are distinguishable from digital signals because the latter always take values only from a finite set of predetermined possibilities, such as the set $\{0\text{V}, 5\text{V}\}$. Analog voltages and currents can be used to control things directly, like the volume of a car radio. In a simple analog radio, a knob is connected to a variable resistor. As the knob is turned, the resistance goes up or down; thus, the current flowing through the resistor increases or decreases. This changes the amount of current driving the speakers, thus increasing or decreasing the volume. An analog circuit is one, like the radio, whose output is linearly proportional to its input. As intuitive and simple as analog control may seem, it is not always economically attractive or otherwise practical. For one thing, analog circuits tend to drift over time and can, therefore, be very difficult to tune. Precision analog circuits, which solve that problem, can be very large, heavy (for example, older home stereo equipment), and expensive. Analog circuits can also get very hot; the power dissipated is proportional to the voltage across the active elements multiplied by the current through them. Analog circuitry can also be sensitive to noise. Because of its infinite resolution, any perturbation or noise on an analog signal neces-



30% duty cycle waveform 70% duty cycle waveform

Fig. 1: Examples of PWM waveforms with different duty cycles.

sarily changes the current value. By controlling analog circuits digitally, system costs and power consumption can be drastically reduced.

PWM finds application in a variety of systems. For example, consider a PWM-controlled brake. To put it simply, a brake is a device that clamps down hard on something. In many brakes, the amount of clamping pressure (or stopping power) is controlled with an analog input signal. The more voltage or current that's applied to the brake, the more pressure the brake will exert. The output of a PWM controller could be connected to a switch between the supply and the brake. To produce more stopping power, the software need only increase the duty cycle of the PWM output. If a specific amount of braking pressure is desired, measurements would need to be taken to determine the mathematical relationship between duty cycle and pressure. (And the resulting formulae or lookup tables would be tweaked for operating temperature, surface wear, and so on.) To set the pressure on the brake to, say, 100psi, the software would do a reverse lookup to determine the duty cycle that should produce that amount of force. It would then set the PWM duty cycle to the new value and the brake would respond accordingly. If a sensor is available in the system, the duty cycle can be tweaked, under closed-loop control, until the desired pressure is precisely achieved.

Consider a DC motor speed control application as another example. A PWM signal is a convenient method to control DC motor speed by varying the average voltage applied to the motor. (The signals are also used to control the direction of motor turns in radio control cars and airplanes.) The average voltage applied to the motor is adjusted hy varying the "on" time of the signal, T_{on}, to the total period of the signal, T_{period}. This quantity, $T_{\text{on}}/T_{\text{period}}$, is called the *duty cycle* of the PWM signal. For example a duty cycle of 70% means that the signal is logic high for 70% of its period and logic low for 30% of its period. The overall effect of this duty cycle is to provide 70% of the motor's source voltage to the motor. Figure 1 shows examples of PWM waveforms with 30% and 70% duty cycles. Generally speaking if V_{max} and V_{min} are the maximum and minimum source voltages, respectively, then the effective voltage provided to the motor is a function of the PWM duty cycle:

$$V_{\text{effective}} = V_{\text{max}} \frac{T_{\text{on}}}{T_{\text{period}}} + V_{\text{min}} \left(1 - \frac{T_{\text{on}}}{T_{\text{period}}}\right)$$

Given a sufficient bandwidth, *any analog value* can be encoded with PWM. Via the use of high-resolution counters, the duty cycle of a square wave is modulated to encode a specific analog signal level. Note that the PWM signal is still digital because, at any given instant of time, the full DC supply is either fully on or fully off. The voltage or current source is supplied to the analog load by means of a repeating series of on and off pulses. The on-time is the time during which the DC supply is applied to the load, and the off-time is the period during which that supply is switched off. Figure 2 shows a simplified view of how a PWM waveform can control a DC motor.

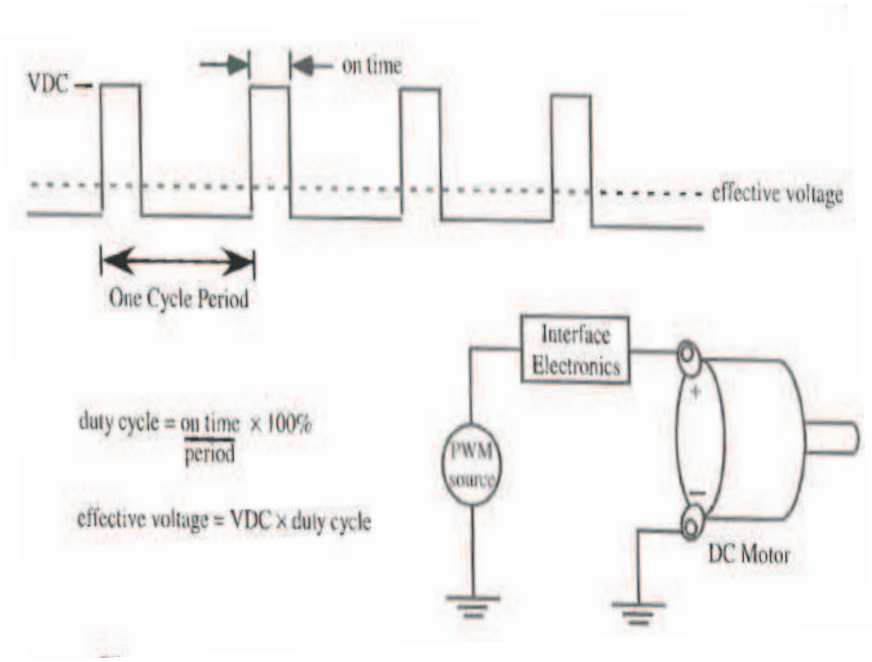


Fig. 2: A PWM signal controlling a DC motor.

Figure 3 shows another example of a simple light-dimming circuit that could be driven using PWM. A 9V battery powers an incandescent light bulb (LB). If we closed the switch connecting the battery and lamp for 50ms, the bulb would receive 9V during that interval. If we then opened the switch for the next 50ms, the bulb would receive 0V. If we repeat this cycle 10 times a second, the bulb will be lit as though it were connected to a 4.5V battery (50% of 9V). We say that the duty cycle is 50% and the modulating frequency is 10Hz. Most loads, inductive and capacitative alike, require a much higher modulating frequency than 10Hz. Imagine that our lamp was switched on for five seconds, then off for five seconds, then on again. The duty cycle would still be 50%, but the bulb would appear brightly lit for the first five seconds and off for the next. In order for the bulb to see a voltage of 4.5 volts, the cycle period must be short relative to the load's response time to a change in the switch state. To achieve the desired effect of a dimmer (but always lit) lamp, it is necessary to increase the modulating frequency. The same is true in other applications of PWM. Common modulating frequencies range from 1kHz to 200kHz.

To generate a PWM signal on a microcontroller, the firmware must perform the following steps:

- Set the period in the on-chip timer/counter that provides the modulating square wave. Let's call this register PWPERx.
- Set the on-time or the duty cycle in the PWM control register. Let's call this register PWDTYx.
- Set the direction of the PWM output, which is one of the general-purpose I/O pins.

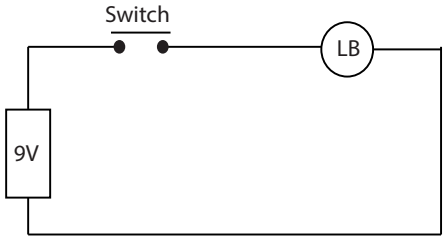


Fig. 3: A light-dimming circuit controlled using PWM signals.

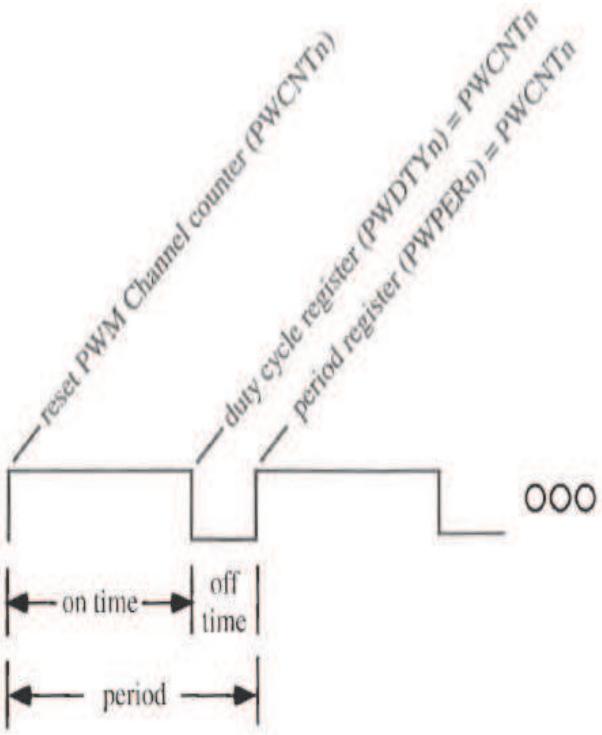


Fig. 4: Generating the PWM signal. The PWM system generates the signal by comparing the duty cycle value to the PWM channel Counter (PWCNTx). Whenever the values in the PWDTYx register and the PWCNTx match the active portion of the signal is de-asserted. The signal will remain de-asserted until the value in the PWCNTx register equal the value in the PWPERx register. The PWMCNTx register is incremented on every clock tick of the PWM clock source.

- Start the timer.
- Enable the PWM controller.

Figure 4 shows the process of generating a PWM signal on the microcontroller. The PWM system generates a signal by comparing the duty cycle value to the PWM Channel Counter, maintained in the register called PWCNTx. When the value in the PWDTYx register and the PWCNTx match, the active portion of the signal is de-asserted. In other words, if the duty cycle was active high, the signal will transition to logic low. The signal will remain low until the value in the PWCNTx register equals the value in the PWPERx register, that is when the end of the PWM period is reached. The PWCNTx register is reset at the beginning of the PWM cycle. The PWCNTx increments on every PWM clock tick. This procedure repeats continually to generate the PWM signal.

The time base for the PWM system is provided by a user-specified clock based on the application of interest—from high frequency, short duty cycle signals to low frequency, long duty cycle signals. With an 8-bit PWM channel, the duty cycle and period counts may range from 0 to 255 ($2^8 - 1$) clock ticks. The 16-bit channels may store counts from 0 to 65,535 ($2^{16} - 1$), allowing for the generation of low frequency signals. Also, the signals may be programmed for left-aligned or center-aligned output.